NERSC Users Group
Monthly Webinar
12/08/2016

Announcements
Agenda

- Cori status and plans for user access
- Allocations and usage update; plans for charging in 2017
- Intel Xeon Phi (KNL) overview and modes of operation
- Shifter overview and hands-on demo
Cori update and user access plans

Richard Gerber
Dec 8, 2016
Cori Update

- Cori Haswell and Xeon Phi (KNL) nodes have been fully integrated into a single system
- Haswell nodes are in full production since Oct. 31
- NESAP teams are on KNL nodes, testing codes and identifying system issues
- KNL nodes testing, debugging is ongoing
  - 1st of a kind processor and network scale pose complex challenges
  - NERSC and Cray staff working with few breaks while creating innovative solutions on a daily basis. Strong support from Intel and SchedMD (SLURM)
  - Connecting 9,300 KNL nodes to network has caused disruptions that also affect Haswell nodes
Cori KNL: Things Look Good!

- System is almost in a steady enough state for use by general NERSC users
- Users will be added in waves to Cori KNL debug queues to start testing codes
  - This will likely start within the next few weeks
- Access to full KNL system will still be restricted to NESAP teams for a number of weeks
- Performance on NESAP codes at small scale is exceeding our expectations
  - Not enough experience yet at large scale to comment
Allocations and usage update; charging plans for 2017

Richard Gerber
Dec 8, 2016
NERSC will once again overdeliver on compute hour commitments to DOE Production and ALCC

**DOE Production**
Target: 2,477 M Hrs
Pace: 2,565 M Hrs

**ALCC**
Target: 223 M Hrs
Delivered: 249 M Hrs

**Scavenger**
Delivered: 78 M Hrs
(3% of total)
We know there is queue pain

**Edison**

Oct. avg wait: 112 hours  
Nov. avg wait: 39 hours  
‘Typical’ wait: ~12-18 hours

**Cori Haswell**

Oct. availability: ~0%  
Nov. availability: 84%  
Target: 90%  
NERSC ‘typical’: 96-99.5%

Queue wait time “heat maps” where each cell is color coded by the average wait time. Hours requested increases across columns and nodes requested increases down rows. Red color indicates a wait time of greater than 36 hours.
In addition to enabling science at unprecedented scale

- Cori KNL nodes will provide an estimated 4.8 B DOE Prod NERSC Hrs/year
- 2X that provided by Edison and Cori Haswell combined
- 3X overall increase in NERSC hours

**KNL nodes will be “free” through June 30, 2017!**

Edison and Cori Haswell will continue to be in full production mode for all of NERSC AY2017.
2017 Allocation Plans

• 2017 allocation decision announcements from DOE program managers scheduled for Friday, Dec. 9
  – May be delayed if not all received and processed on time
• 2.4 B NERSC hours to be distributed on Dec. 9 for use in AY 2017 (starts Jan. 10, 2017)
  – Minus reserves held by program managers
• Additional 2.4 B hours for use on KNL will be distributed in spring 2017
  – For use in 2017 after KNL charging begins on July 1, 2017
  – DOE program manager decisions will be based on applications’ readiness for KNL
  – NERSC will provide process for demonstrating readiness
Preparing for Cori KNL access

Steve Leak
Dec 8, 2016
Topics

• How is KNL different to Haswell?
  – Faster and slower
  – MCDRAM and clustering modes
• How to compile for KNL
• How to submit to KNL nodes
Key Messages

• Don’t Panic!
  – Using KNL is *mostly* the same as you already use Cori

• Recompile
  – module swap craype-haswell craype-mic-knl

• Submit with:
  – #SBATCH -C knl,cache,flat

• See notes (downloads) at:
KNL vs Haswell

- KNL can run Haswell executables
KNL vs Haswell

But ..

- Haswell Executables can’t fully use KNL hardware

AVX2 (haswell)
Operation on 4 DP words

AVX-512 (knl)
Hardware can compute 8 DP words per instruction
And ..

- KNL relies more on vectorization for performance
KNL vs Haswell

And ..

- KNL has a lot more cores (and even more hyperthreads)

68 cores (2 per tile)
4 threads / core
= 272 threads
KNL vs Haswell

And ..

• KNL has MCDRAM
MCDRAM/Clustering Modes

You’ll hear a lot about these …

• Quad, flat, cache, snc, a2a, …

2 “knobs” to turn

• MCDRAM mode: “cache”, “flat”, “hybrid”
• Clustering mode: “quad”, “snc4”, “snc2”, …
MCDRAM in a nutshell

- **16GB on-chip memory**
  - cf 96GB off-chip DDR (Cori)
- **Not (exactly) a cache**
  - Latency similar to DDR
- **But very high bandwidth**
  - ~5x DDR
- **2 ways to use it:**
  - “Cache” mode: invisible to OS, memory pages are cached in MCDRAM (cache-line granularity)
  - “Flat” mode: appears to OS as separate NUMA node, with no local CPUs. Accessible via numactl, libnuma (page granularity)
MCDRAM modes

Cache mode

- DDR
- MCDRAM
- Cores & cache

“memory”

Cache mode:
- No need to do anything
- *Most* of the flat-mode performance, *most* of the time

Flat mode

- DDR
- “memory 0”
- Cores & cache
- MCDRAM

- Which memory do I want to use?
- For which arrays?

Flat mode: better achievable performance

Hybrid mode: a bit of each
Clustering Modes

Cache coherency
- does another cache hold this memory address?

“Tag directory”
- list of address:cache-id
- Distributed tag directory, each looks after part of the address

On local cache miss:
- Check tag-directory-part for other-cache-id
- Fetch from other-cache (or memory)
- Takes time! (latency)
Clustering modes

• **Quadrant (“quad”)**
  – Distribute tag directories so each manages only memory nearby
  – Invisible to user, but improves memory access latency

• **Sub-NUMA-clustering (“snc4”)**
  – Expose each quadrant and its managed memory as a NUMA node
  – Even lower latency, but less flexible

• **We recommend quadrant mode for most users**
Topics

• How is KNL different to Haswell?
  – Faster and slower
  – MCDRAM and clustering modes

• How to compile for KNL

• How to submit to KNL nodes
How to compile

Best: Use Cray compiler wrappers

`module swap craype-haswell craype-mic-knl`

- The loaded craype-* module sets the target that the compiler wrappers (cc, CC, ftn) build for
  - Eg `-mkn1` (GNU compiler), `-hmic-knl` (Cray compiler)
- `craype-haswell` is default on login nodes
- `craype-mic-knl` is for KNL nodes
Topics

• How is KNL different to Haswell?
  – Faster and slower
  – MCDRAM and clustering modes
• How to compile for KNL
• How to submit to KNL nodes
How to run on KNL

#SBATCH -C knl,quad,cache

- Tells Slurm you want to run on this type of node
- If insufficient nodes in that configuration, will reboot some to change mode
  - This takes time!
- Which modes are there now?
- `sinfo -p knl --format="%15b %8D %A"
- See notes (doanloads) at:
  - (affinity etc)
Key Messages

• Don’t Panic!
  – Using KNL is *mostly* the same as you already use Cori

• Recompile
  – module swap craype-haswell craype-mic-knl

• Submit with:
  – #SBATCH -C knl,cache,flat

• See notes (downloads) at: